This listing of claims will replace all prior versions, and listings, of claims in the application.

Listing of Claims:

1. (Previously Presented) A method for controlling the computational resources of at least one coprocessor in a host computing system having a host processor, comprising:

controlling the at least one coprocessor of the computing system with commands of command buffers submitted to the at least one coprocessor by a host processor of the host computing system;

transmitting, by the at least one coprocessor, data back to the host computing system in response to commands in at least one command buffer of the command buffers;

scheduling the transmission of the commands of the command buffers by a managing object included in the host computing system,

wherein:

the computational resources of the at least one coprocessor are simultaneously available to a plurality of applications instantiated on the host computing system; and

said managing object allows a plurality of types of coprocessor context; and

affiliating coprocessor context with a host processor thread context.

2. (Previously Presented) A method according to claim 1, wherein said scheduling includes scheduling the transmission of the commands of the command buffers by an operating system included in the host computing system.

3. (Previously Presented) A method according to claim 1, wherein the managing object is notified by a coprocessor that commands of a command buffer has finished execution.

4. (Previously Presented) A method according to claim 1, further including queuing commands of a new command buffer for a coprocessor to begin executing when commands of a current command buffer are finished.

5. (Previously Presented) A method according to claim 1, further including specifying a coprocessor context switch when commands of a command buffer is submitted.

6-7. (Canceled)

8. (Previously Presented) A method according to claim 1, further including integrating by the managing object the context switching code for the host processor and the coprocessor.

9. (Previously Presented) A method according to claim 1, further including notifying the managing object by a coprocessor that commands of a command buffer are invalid.

10. (Original) A method according to claim 1, further including resetting a coprocessor of the at least one coprocessor if the coprocessor is unresponsive for a predetermined period of time.

11. (Previously Presented) A method according to claim 1, further including translating by a hardware-specific driver object, via an application programming interface of the managing object, instructions of commands of a command buffer into hardware-specific instructions during composition of the commands of the command buffer.

12. (Original) A method according to claim 11, wherein said translating runs in user mode.

13. (Previously Presented) A method according to claim 12, further including allocating a guard page at the end of the commands of the command buffer to facilitate efficient detection of buffer overflow.

14. (Original) A method according to claim 12, wherein the user mode driver and corresponding runtime component are provided in intermediate language form and the method further includes just in time (JIT) compiling on a client device having the user mode driver and runtime component.

15. (Previously Presented) A method according to claim 14, wherein the application is also provided in intermediate language form and said JIT compiling includes JIT compiling the application on the client device with at least the user mode driver.

16. (Previously Presented) A method according to claim 12, wherein said driver object coordinates with a corresponding kernel mode driver object to edit the commands of the command buffer before submission to hardware.

17. (Original) A method according to claim 1, wherein the at least one coprocessor includes at least one graphics processing unit.

18. (Original) A method according to claim 1, further including preempting by the at least one coprocessor upon the occurrence of an external event.

19. (Original) A method according to claim 18, wherein the external event is the operating system making a call to a corresponding kernel mode driver object to preempt the at least one coprocessor.

20. (Original) A method according to claim 18, wherein the host processor is interrupted to coordinate scheduling of processing time.

21. (Previously Presented) A method according to claim 1, further including virtualizing by the managing object at least one resource of the at least one coprocessor during editing of ~~the~~ control data streams of a command buffer before transmission to a coprocessor.

22. (Original) A method according to claim 21, wherein the at least one resource virtualized by the managing object of the at least one coprocessor is memory.

23. (Previously Presented) A method according to claim 1, wherein the managing object uses thread synchronization primitives to coordinate construction, scheduling and transmission of the commands of coprocessor command buffers.

24-26. (Canceled)

27. (Previously Presented) At least one tangibly embodied computer readable medium having stored thereon a plurality of computer-executable modules for controlling the computational resources of at least one coprocessor in a host computing system having a host processor, the computer executable modules comprising:

a managing object for controlling the at least one coprocessor of the computing system with command data of command buffers submitted to the at least one coprocessor by a host processor of the host computing system and for scheduling the transmission of the command data; and

means for transmitting, by the at least one coprocessor, data back to the host computing system in response to command data of at least one command buffer of the command buffers;

whereby the computational resources of the at least one coprocessor are simultaneously available to a plurality of applications instantiated on the host computing system; wherein:

said managing object allows a plurality of types of coprocessor context; and

said managing object affiliates coprocessor context with a host processor thread context.

28. (Original) At least one computer readable medium according to claim 27, wherein said managing object is included in the operating system of the host computing system.

29. (Previously Presented) At least one computer readable medium according to claim 27, wherein the managing object is notified by a coprocessor that command data of a command buffer has finished execution.

30. (Previously Presented) At least one computer readable medium according to claim 27, wherein said managing object queues command data of a new command buffer for a coprocessor to begin executing when command data of a current command buffer is finished.

31. (Previously Presented) At least one computer readable medium according to claim 27, wherein said managing object specifies a coprocessor context switch when command data of a command buffer is submitted.

32-33. (Canceled)

34. (Previously Presented) At least one computer readable medium according to claim 27, wherein said managing object integrates the context switching code for the host processor and the coprocessor.

35. (Previously Presented) At least one computer readable medium according to claim 27, wherein a coprocessor comprises means for notifying the managing object that command data of a command buffer is invalid.

36. (Original) At least one computer readable medium according to claim 27, wherein said managing object resets a coprocessor of the at least one coprocessor if the coprocessor is unresponsive for a predetermined period of time.

37. (Previously Presented) At least one computer readable medium according to claim 27, wherein said managing object enables translating by a hardware-specific driver object instructions of command data of a command buffer into hardware-specific instructions during composition of command data of the command buffer.

38. (Original) At least one computer readable medium according to claim 37, wherein said translating by the driver object runs in user mode.

39. (Previously Presented) At least one computer readable medium according to claim 38, wherein said managing object allocates a guard page at the end of the command data of the command buffer to facilitate efficient detection of buffer overflow.

40. (Original) At least one computer readable medium according to claim 38, wherein the user mode driver and corresponding runtime component are provided in intermediate language form and the user mode driver and runtime component are just in time (JIT) compiled.

41. (Previously Presented) At least one computer readable medium according to claim 40, wherein the application is also provided in intermediate language form and at least the application and the user mode driver are JIT compiled.

42. (Previously Presented) At least one computer readable medium according to claim 38, wherein said driver object coordinates with a corresponding kernel mode driver object to edit the command data of the command buffer before submission to hardware.

43. (Original) At least one computer readable medium according to claim 27, wherein the at least one coprocessor includes at least one graphics processing unit.

44. (Original) At least one computer readable medium according to claim 27, wherein the managing object is preempted by the at least one coprocessor upon the occurrence of an external event.

45. (Original) At least one computer readable medium according to claim 44, wherein the external event is the operating system making a call to a corresponding kernel mode driver object to preempt the at least one coprocessor.

46. (Original) At least one computer readable medium according to claim 27, wherein the host processor is interrupted to coordinate scheduling of processing time by the managing object.

47. (Previously Presented) At least one computer readable medium according to claim 27, wherein the managing object virtualizes at least one resource of the at least one coprocessor during editing of control data streams of a command buffer before transmission to a coprocessor.

48. (Original) At least one computer readable medium according to claim 47, wherein the at least one resource virtualized by the managing object of the at least one coprocessor is memory.

49. (Previously Presented d) At least one computer readable medium according to claim 27, wherein the managing object uses thread synchronization primitives to coordinate construction, scheduling and transmission of command data of coprocessor command buffers.

50-51. (Canceled)

52. (Previously Presented) A computing device having a host processor for controlling the computational resources of at least one coprocessor in a host computing system, comprising:

a managing object for controlling the at least one coprocessor of the host computing system with command data of command buffers submitted to the at least one coprocessor by the host processor of the host computing system and for scheduling the transmission of the command data of the command buffers; and means for transmitting, by the at least one coprocessor, data back to the host computing system in response to command data in at least one command buffer of the command buffers;

whereby the computational resources of the at least one coprocessor are simultaneously available to a plurality of applications instantiated on the host computing system; wherein:

said managing object allows a plurality of types of coprocessor context; and

said managing object affiliates coprocessor context with a host processor thread context.

53. (Original) A computing device according to claim 52, wherein said managing object is included in the operating system of the host computing system.

54. (Previously Presented) A computing device according to claim 52, wherein the managing object is notified by a coprocessor that command data of a command buffer has finished execution.

55. (Previously Presented) A computing device according to claim 52, wherein said managing object queues command data of a new command buffer for a coprocessor to begin executing when command data of a current command buffer is finished.

56. (Previously Presented) A computing device according to claim 52, wherein said managing object specifies a coprocessor context switch when command data of a command buffer is submitted.

57-58. (Canceled)

59. (Previously Presented) A computing device according to claim 52, wherein said managing object integrates the context switching code for the host processor and the coprocessor.

60. (Previously Presented) A computing device according to claim 52, wherein a coprocessor comprises means for notifying the managing object that command data of a command buffer is invalid.

61. (Original) A computing device according to claim 52, wherein said managing object resets a coprocessor of the at least one coprocessor if the coprocessor is unresponsive for a predetermined period of time.

62. (Previously Presented) A computing device according to claim 52, wherein said managing object enables translating by a hardware-specific driver object instructions of command data of a command buffer into hardware-specific instructions during composition of the command data of the command buffer.

63. (Original) A computing device according to claim 62, wherein said translating by the driver object runs in user mode.

64. (Previously Presented) A computing device according to claim 63, wherein said managing object allocates a guard page at the end of the command data of the command buffer to facilitate efficient detection of buffer overflow.

65. (Original) A computing device according to claim 63, wherein the user mode driver and corresponding runtime component are provided in intermediate language form and the user mode driver and runtime component are just in time (JIT) compiled.

66. (Previously Presented) A computing device according to claim 65, wherein the application is also provided in intermediate language form and at least the application and the user mode driver are JIT compiled.

67. (Previously Presented) A computing device according to claim 63, wherein said driver object coordinates with a corresponding kernel mode driver object to edit the command data of the command buffer before submission to hardware.

68. (Original) A computing device according to claim 52, wherein the at least one coprocessor includes at least one graphics processing unit.

69. (Original) A computing device according to claim 52, wherein the managing object is preempted by the at least one coprocessor upon the occurrence of an external event.

70. (Original) A computing device according to claim 69, wherein the external event is the operating system making a call to a corresponding kernel mode driver object to preempt the at least one coprocessor.

71. (Original) A computing device according to claim 52, wherein the host processor is interrupted to coordinate scheduling of processing time by the managing object.

72. (Previously Presented) A computing device according to claim 52, wherein the managing object virtualizes at least one resource of the at least one coprocessor during editing of control data streams of a command buffer before transmission to a coprocessor.

73. (Original) A computing device according to claim 72, wherein the at least one resource virtualized by the managing object of the at least one coprocessor is memory.

74. (Previously Presented) A computing device according to claim 52, wherein the managing object uses thread synchronization primitives to coordinate construction, scheduling and transmission of command data of coprocessor command buffers.